Course Outcome 1

Experiment 1 **Date:**

Basic Java Programs

Aim:

Write the following programs
i) Print the prime numbers up to a limit

Program

```
import java.io.*;
class PrimeInLimit
{
public static void main(String args[]) throws IOException
{
DataInputStream x=new DataInputStream(System.in);
System.out.println("Enter Limit");
int n=Integer.parseInt(x.readLine());
System.out.println("Prime numbers up to "+n+":");
for(int num = 2; num <= n; num++)
{
int flag=0;
for (int i = 2; i<num/2; i++)
{
if (num % i == 0)
{
flag=1;
break;
}
}
if(flag==0)
{
System.out.println(num);
}
}
}
}
```

Output

```
mits@mits-Veriton-M200-H510:~/gokul java$ java PrimeInLimit
Enter Limit
15
Prime numbers up to 15:
2
3
4
5
7
11
13
```

ii) Print the 3-digit Armstrong numbers between two intervals.

Program

```
import java.io.*;
class ArmstrongInLimit
{
public static void main(String args[]) throws IOException
{
DataInputStream x=new DataInputStream(System.in);
System.out.println("Enter Limit 1");
int num1=Integer.parseInt(x.readLine());
System.out.println("Enter Limit 2");
int num2=Integer.parseInt(x.readLine());
System.out.println("Armstrong Numbers:");
for (int i = num1; i<num2; i++)
{
int n=0;
int temp=i;
while (temp != 0)
{
temp=temp/10;
n=n+1;
}
int sum=0;
temp=i;
while (temp != 0)
{
```

```
int digit=temp%10;
sum=sum+(int)Math.pow(digit,n);
temp=temp/10;
}
if (sum==i)
{
System.out.println(i);
}
}
}
}
```

Output

```
mits@mits-Veriton-M200-H510:~/gokul java$ java ArmstrongInLimit
Enter Limit 1:
100
Enter Limit 2:
500
Armstrong Numbers:
153
370
371
407
```

Experiment 2 Date:

One-Dimensional Array

Aim:

Write a Java program to search an element in an array

Program

```
import java.io.*;
class ElementCheck
{
public static void main(String args[]) throws IOException
{
DataInputStream x=new DataInputStream(System.in);
System.out.println("enter limit of array");
int n=Integer.parseInt(x.readLine());
int a[]=new int[n];
System.out.println("enter elements");
for(int i=0;i<n;i++)
{
a[i]=Integer.parseInt(x.readLine());
}
System.out.println("elements");
for(int i=0;i<n;i++)
{
System.out.print(a[i]+" ");
}
System.out.println();
int c=1,flag=0;
System.out.println("enter element to check");
int y=Integer.parseInt(x.readLine());
for(int i=0;i<n;i++)
{
if(a[i]==y)
{
flag=1;
break;
}
c=c+1;
}
if(flag==1)
```

```
{
System.out.println("element found at position "+c);
}
else
{
System.out.println("element not found");
}
}
}
```

Output

```
mits@mits-Veriton-M200-H510:~/gokul java$ java ElementCheck
enter limit of array
4
enter elements
8
4
6
2
elements
8 4 6 2
enter element to check
6
element found at position 3


mits@mits-Veriton-M200-H510:~/gokul java$ java ElementCheck
enter limit of array
4
enter elements
8
4
6
2
elements
8 4 6 2
enter element to check
10
element not found
```

Experiment 3 Date:

Two-Dimensional Array

Aim:

Write a program to read a matrix from the console and check whether it is symmetric or not.

Program

```
import java.io.*;
class SymmetricMatrix
{
public static void main(String args[]) throws IOException
{
int flag=0;
DataInputStream x=new DataInputStream(System.in);
System.out.println("Enter order of matrix");
int n=Integer.parseInt(x.readLine());
int a[][]=new int[n][n];
System.out.println("Enter elements of Matrix");
for(int i=0;i<n;i++)
{
for(int j=0;j<n;j++)
{
a[i][j]=Integer.parseInt(x.readLine());
}
}
System.out.println("Matrix elements");
for(int i=0;i<n;i++)
{
for(int j=0;j<n;j++)
{
System.out.print(a[i][j]+" ");
}
System.out.println();
}
for(int i=0;i<n;i++)
{
for(int j=0;j<n;j++)
{
if(a[i][j]!=a[j][i])
```

```
{
flag=1;
break;
}
}
}
if(flag==0)
{
System.out.println("Matrix is Symmetric");
}
else
{
System.out.println("Matrix is not Symmetric");
}
}
}
```

Output

```
mits@mits-Veriton-M200-H510:~/gokul java$ java SymmetricMatrix
Enter order of matrix
3
Enter elements of Matrix
1
0
1
0
1
0
1
0
1
Matrix elements
1 0 1
0 1 0
1 0 1
Matrix is Symmetric

mits@mits-Veriton-M200-H510:~/gokul java$ java SymmetricMatrix
Enter order of matrix
3
```

Enter elements of Matrix
1
0
4
5
1
6
1
0
5
Matrix elements
1 0 4
5 1 6
1 0 5
Matrix is not Symmetric

Experiment 4 **Date:**

String Handling Methods- 1

Aim:

Perform the following operations on strings
i. Find the length of the string
ii. Character at second and fourth position
iii. Find the sub string using start index only
iv. Find the sub string using start index and end index
v. Compare two strings lexicographically.
vi. Compare two strings lexicographically, ignoring case differences.
vii. Concatenate a given string to the end of another string.
viii. Replace a specified character with another character.
ix. Check whether a given string starts with another string.
x. Convert all characters in a string to lowercase
xii. Convert all characters in a string to uppercase.

Program

```
import java.io.*;
class StringOperations
{
public static void main(String args[]) throws IOException
{
DataInputStream x=new DataInputStream(System.in);
System.out.println("1.Length of string");
System.out.println("Enter a string");
String s=x.readLine();
System.out.println("string is: "+s);
System.out.println("length of string is: "+s.length());
System.out.println();

System.out.println("2.Character At Position");
System.out.println("Character at second position: "+s.charAt(1));
System.out.println("Character at fourth position: "+s.charAt(3));
System.out.println();

System.out.println("3.Substring Using Start Index");
System.out.println("Enter start index");
int st=Integer.parseInt(x.readLine());
```

```
System.out.println("Substring from start index: "+s.substring(st));
System.out.println();

System.out.println("4.Substring Using Start and End Index");
System.out.println("Enter start index");
int st1=Integer.parseInt(x.readLine());
System.out.println("Enter end index");
int ed=Integer.parseInt(x.readLine());
System.out.println("Substring from start to end index: "+s.substring(st1,ed));
System.out.println();

System.out.println("5.Compare Strings");
System.out.println("Enter a new string1");
String s8=x.readLine();
System.out.println("Enter a new string2");
String s9=x.readLine();
if(s8.equals(s9))
{
System.out.println("String equal");
}
else
{
System.out.println("String not equal");
}
System.out.println();

System.out.println("6.Compare Strings(Ignore Case)");
System.out.println("Enter a new string1");
String s10=x.readLine();
System.out.println("Enter a new string2");
String s11=x.readLine();
if(s10.equalsIgnoreCase(s11))
{
System.out.println("String equal");
}
else
{
System.out.println("String not equal");
}
System.out.println();
```

```
System.out.println("7.Concatenate Strings");
System.out.println("Enter a new string1");
String s1=x.readLine();
System.out.println("Enter a new string2");
String s2=x.readLine();
System.out.println("After Concatenate: "+s1.concat(s2));
System.out.println();

System.out.println("8.Character Replace");
System.out.println("Enter a new string");
String s3=x.readLine();
System.out.println("Enter a character to replace");
char ch1=(x.readLine().charAt(0));
System.out.println("Enter new character");
char ch2=(x.readLine().charAt(0));
System.out.println("After Replace: "+s3.replace(ch1,ch2));
System.out.println();

System.out.println("9.Start With a String");
System.out.println("Enter a new string");
String s4=x.readLine();
System.out.println("Enter start string");
String s5=x.readLine();
if(s4.startsWith(s5))
{
System.out.println("String start with "+s5);
}
else
{
System.out.println("String not start with "+s5);
}
System.out.println();

System.out.println("10.Uppercase");
System.out.println("Enter a new string");
String s6=x.readLine();
System.out.println("Uppercase: "+s6.toUpperCase());
System.out.println();

System.out.println("11.Lowercase");
```

```
System.out.println("Enter a new string");
String s7=x.readLine();
System.out.println("Lowercase: "+s7.toLowerCase());
}
}
```

Output

```
mits@mits-Veriton-M200-H510:~/gokul java$ java StringOperations
1.Length of string
Enter a string
gokulrajc
string is:
length of string is: 9

2.Character At Position
Character at second position: o
Character at fourth position: u

3.Substring Using Start Index
Enter start index
4
Substring from start index:
ulrajc

4.Substring Using Start and End Index
Enter start index
2
Enter end index
6
Substring from start to end index:
Okul

5.Compare Strings
Enter a new string1
abcd
Enter a new string2
ABCD
String not equal

6.Compare Strings(Ignore Case)
```

Enter a new string1
abcd
Enter a new string2
ABCD
String equal

7.Concatenate Strings
Enter a new string1
gokul
Enter a new string2
raj
After Concatenate:
gokulraj

8.Character Replace
Enter a new string
malayalam
Enter a character to replace
m
Enter new character
x
After Replace:
xalayalax

9.Start With a String
Enter a new string
hi welcome
Enter start string
hi
String start with hi

10.Uppercase
Enter a new string
abcd
Uppercase: ABCD

11.Lowercase
Enter a new string
ABCD
Lowercase: abcd

Experiment 5 Date:

String Handling Methods- 2

Aim:

Write a java program to

i. Check whether a given string is palindrome or not.

Program

```
import java.io.*;
class StringPallindrome
{
public static void main(String args[]) throws IOException
{
DataInputStream x=new DataInputStream(System.in);
System.out.println("Enter a string");
String s1=x.readLine();
String s2="";
System.out.println("String:"+s1);
int l = s1.length();
for(int i=l-1;i>=0;i--)
{
s2=s2+s1.charAt(i);
}
System.out.println("Reversed String:"+s2);
if(s1.equals(s2))
{
System.out.println("pallindrome");
}
else
{
System.out.println("not pallindrome");
}
}
}
```

Output

```
mits@mits-Veriton-M200-H510:~/gokul java$ java StringPallindrome
Enter a string
malayalam
```

```
String:malayalam
Reversed String:malayalam
pallindrome

mits@mits-Veriton-M200-H510:~/gokul java$ java StringPallindrome
Enter a string
welcome
String:welcome
Reversed String:emoclew
not pallindrome
```

ii. Sorting a given list of names in ascending order

Program

```
import java.io.*;
class NameSort
{
public static void main(String args[]) throws IOException
{
DataInputStream x=new DataInputStream(System.in);
System.out.println("Enter limit");
int n=Integer.parseInt(x.readLine());
String str[]=new String[n];
String temp;
System.out.println("Enter names");
for(int i=0;i<n;i++)
{
str[i]=x.readLine();
}
for (int i=0;i<n;i++)
{
for (int j=0;j<n;j++)
{
if(str[i].compareTo(str[j]) > 0)
{
temp=str[i];
str[i]=str[j];
str[j]=temp;
}
}
```

```
}
System.out.println();
System.out.println("Names");
for(int i=0;i<n;i++)
{
System.out.println(str[i]);
}
}
}
```

Output

```
mits@mits-Veriton-M200-H510:~/gokul java$ java NameSort
Enter limit
5
Enter names
thomas
abhijith
allen
gokul
adwaith

Names
abhijith
adwaith
allen
gokul
thomas
```

Experiment 6 Date:

StringBuffer Class Methods

Aim:

Write a program in java for string handling which performs the following
i. Check the capacity of the StringBuffer object.
ii. Reverse the content of this string and convert the resultant string in upper case
iii. Read another string and append it to the resultant string of above.

Program

```
import java.io.*;
class StringBufferExample
{
public static void main(String args[]) throws IOException
{
DataInputStream d = new DataInputStream(System.in);
System.out.println("Enter a string:");
String str = (d.readLine());
StringBuffer s = new StringBuffer(str);
System.out.println("Capacity is "+s.capacity());
s.reverse();
String s2 = s.toString().toUpperCase();
StringBuffer ss = new StringBuffer(s2);
System.out.println("After resversing and converting to uppercase: "+ss);
System.out.println("Enter a string to append:");
String s1 = (d.readLine());
System.out.println("New String: "+ss.append(s1));
}
}
```

Output

```
mits@mits-Veriton-M200-H510:~/gokul java$ java StringBufferExample
Enter a string:
gokul
Capacity is 21
After resversing and converting to uppercase: LUKOG
Enter a string to append:
raj
New String: LUKOGraj
```

Course Outcome 2

Experiment 7 **Date:**

Initialize instance variables using class and method

Aim:

Program to demonstrate use of command line arguments to initialize values to member variables in a class and to display them.

Hint:- Create a class containing Rlno, stud_name, engmark, mathsmark, totalmark. While executing the program we have to pass arguments through command line. These values are obtained in an array which is passed as argument to main function, here it is args[ ]. The marks are converted correspondingly and then passed to constructor where values are stored to class variables. Find the total marks and later displayed using display function.

Program

```
class Student
{
int rollno;
String name;
int eng;
int math;
int total;
Student(int r,String s,int e,int m)
{
rollno=r;
name=s;
eng=e;
math=m;
}
void totalmark()
{
total=eng+math;
}
void display()
{
System.out.println("roll no: "+rollno);
System.out.println("name: "+name);
System.out.println("english mark: "+eng);
System.out.println("maths mark: "+math);
```

```
System.out.println("total mark: "+total);
}
}
class TotalMark
{
public static void main(String args[])
{
int r=Integer.parseInt(args[0]);
String s=args[1];
int e=Integer.parseInt(args[2]);
int m=Integer.parseInt(args[3]);
Student s1=new Student(r,s,e,m);
s1.totalmark();
s1.display();
}
}
```

Output

```
mits@mits-Veriton-M200-H510:~/gokul java$ java TotalMark 29 gokul 60 70
roll no: 29
name: gokul
english mark: 60
maths mark: 70
total mark: 130
```

Experiment 8 Date:

Initialize instance variables inside the class using constructor

Aim:

Program to demonstrate use of constructors to initialize values to member variables in a class and to display them.

Hint:- empno , empname and salary are the class members of the class employee1. From the main function we are passing the values directly to a constructor, the constructor initializes the values to member variables. The display function is used to display the stored values of the member variables.

Program

```
import java.io.*;
class Employee
{
int empno;
String empname;
int salary;
Employee(int r,String n,int s)
{
empno=r;
empname=n;
salary=s;
}
void display()
{
System.out.println("employee details");
System.out.println("employee no: "+empno);
System.out.println("employee name: "+empname);
System.out.println("salary: "+salary);
}
}
class EmployeeDetails
{
public static void main(String args[]) throws IOException
{
DataInputStream x=new DataInputStream(System.in);
System.out.println("enter employee no");
int r=Integer.parseInt(x.readLine());
```

```
System.out.println("enter employee name");
String n=x.readLine();
System.out.println("enter employee salary");
int s=Integer.parseInt(x.readLine());
Employee e1=new Employee(r,n,s);
e1.display();
}
}
```

Output

```
mits@mits-Veriton-M200-H510:~/gokul java$ java EmployeeDetails
enter employee no
101
enter employee name
Gokul raj c
enter employee salary
25000
employee deatils
employee no: 101
employee name: Gokul raj c
salary: 25000
```

Experiment 9 Date:

Matrix Operations

Aim:

Read 2 matrices from the console and perform matrix addition and multiplication using class and object.

Program

```
import java.io.*;
class Matrix
{
int row;
int cols;
int arr[][];
int arr1[][];
int arr2[][];
Matrix(int r,int c)
{
row=r;
cols=c;
arr=new int[r][c];
}

void readMatrix(DataInputStream x) throws IOException
{
for(int i=0;i<row;i++)
{
for(int j=0;j<cols;j++)
{
arr[i][j]=Integer.parseInt(x.readLine());
}
}
}

void displayMatrix()
{
for(int i=0;i<row;i++)
{
for(int j=0;j<cols;j++)
```

```
{
System.out.print(arr[i][j]+" ");
}
System.out.println();
}
}

void addMatrix(Matrix other)
{
if((row != other.row) || (cols != other.cols))
{
System.out.println("addition not possible");
}
else
{
arr1=new int[row][cols];
for(int i=0;i<row;i++)
{
for(int j=0;j<cols;j++)
{
arr1[i][j]=arr[i][j]+other.arr[i][j];
System.out.print(arr1[i][j] +" ");
}
System.out.println();
}
}
}

void mulMatrix(Matrix other)
{
if(other.row != other.cols)
{
System.out.println("multiplication not possible");
}
else
{
arr2=new int[row][other.cols];
for(int i=0;i<row;i++)
{
for(int j=0;j<other.cols;j++)
```

```
{
for(int k=0;k<cols;k++)
{
arr2[i][j]=arr2[i][j]+(arr[i][k]*other.arr[k][j]);
}
System.out.print(arr2[i][j] +" ");
}
System.out.println();
}
}
}

}

class MatrixAddMul
{
public static void main(String args[]) throws IOException
{
DataInputStream x = new DataInputStream(System.in);
System.out.println("enter row of matrix1:");
int r1=Integer.parseInt(x.readLine());
System.out.println("enter column of matrix1:");
int c1=Integer.parseInt(x.readLine());
Matrix m1 = new Matrix(r1,c1);
System.out.println("enter values of matrix1:");
m1.readMatrix(x);

System.out.println("enter row of matrix2:");
int r2=Integer.parseInt(x.readLine());
System.out.println("enter column of matrix2:");
int c2=Integer.parseInt(x.readLine());
Matrix m2 = new Matrix(r2,c2);
System.out.println("enter values of matrix1:");
m2.readMatrix(x);

System.out.println("matrix1:");
m1.displayMatrix();
System.out.println("matrix2:");
m2.displayMatrix();
System.out.println("matrix addition:");
```

```
m1.addMatrix(m2);
System.out.println("matrix multiplication:");
m1.mulMatrix(m2);
}
}
```

Output

```
mits@mits-Veriton-M200-H510:~/gokul java$ java MatrixAddMul
enter row of matrix1:
2
enter column of matrix1:
2
enter values of matrix1:
1
2
3
4
enter row of matrix2:
2
enter column of matrix2:
2
enter values of matrix1:
5
6
7
8
matrix1:
1 2
3 4
matrix2:
5 6
7 8
matrix addition:
6 8
10 12
matrix multiplication:
19 22
43 50
```

Experiment 10 Date:

Complex Number Addition

Aim:

Write a Java program to add to complex numbers using object as argument

Program

```
import java.io.*;
class Complex
{
int real;
int imag;
Complex(int r,int i)
{
real = r;
imag = i;
}
void addNumber(Complex other)
{
int real1;
int imag1;
real1=real+other.real;
imag1=imag+other.imag;
System.out.println(real1 + " + " + imag1 + "i");
}

void display()
{
System.out.println(real + " + " + imag + "i");
}
}

public class ComplexAddition
{
public static void main(String args[]) throws IOException
{
int a1,a2,b1,b2;
DataInputStream x=new DataInputStream(System.in);
System.out.println("Complex number 1");
```

```
System.out.println("Enter complex parts:");
a1= Integer.parseInt(x.readLine());
System.out.println("Enter imaginary parts:");
b1= Integer.parseInt(x.readLine());
Complex c1=new Complex(a1,b1);
System.out.println("Complex number 2");
System.out.println("Enter complex parts:");
a2= Integer.parseInt(x.readLine());
System.out.println("Enter imaginary parts:");
b2= Integer.parseInt(x.readLine());
Complex c2=new Complex(a2,b2);

System.out.println("Complex number 1");
c1.display();
System.out.println("Complex number 2");
c2.display();
System.out.println("Complex number addition");
c1.addNumber(c2);
}
}
```

Output

```
mits@mits-Veriton-M200-H510:~/gokul java$ java ComplexAddition
Complex number 1
Enter complex parts:
2
Enter imaginary parts:
3
Complex number 2
Enter complex parts:
4
Enter imaginary parts:
5
Complex number 1
2 + 3i
Complex number 2
4 + 5i
Complex number addition
6 + 8i
```

Experiment 11 Date:

Class and Objects

Aim:

Define a class ‘product’ with data members pcode, pname and price. Create 3 objects of the class and find the product having the lowest price.

Program

```
class Product
{
int price;
String pcode, pname;
Product(String code, String name, int pri)
{
pcode = code;
pname = name;
price = pri;
}
void display()
{
System.out.println("Code: " +pcode);
System.out.println("Name: " +pname);
System.out.println("Price: " +price);
}
}

class ProductDetails
{
public static void main(String args[])
{
Product p1 = new Product("p1", "Mobile", 13000);
Product p2 = new Product("p2", "Watch", 6500);
Product p3 = new Product("p3", "TV", 16000);
System.out.println("Product with the lowest price");
if (p1.price < p2.price && p1.price < p3.price)
{
p1.display();
}
else if (p2.price < p3.price)
```

```
{
p2.display();
}
else
{
p3.display();
}
}
}
```

Output

```
mits@mits-Veriton-M200-H510:~/gokul java$ java ProductDetails
Product with the lowest price
Code: p2
Name: Watch
Price: 6500
```

Experiment 12 Date:

Inner class and Static nested class

Aim:

Create CPU with attribute price. Create inner class Processor with attributes no. of cores, manufacturer and static nested class RAM with attributes memory and manufacturer. Create an object of CPU class and print information of Processor and RAM.

Program

```
import java.util.*;
class CPU
{
int price;
CPU(int price)
{
this.price = price;
}
void display()
{
System.out.println("CPU Info:");
System.out.println("CPU Price:" +price+ " Rs");
}

class Processor
{
int cores;
String manufacturer;
Processor(int cores, String manufacturer)
{
this.cores = cores;
this.manufacturer = manufacturer;
}
void displayProcessorInfo()
{
System.out.println("Processor Info:");
System.out.println("Cores: " + cores);
System.out.println("Manufacturer: " + manufacturer);
}
}
```

```
static class RAM
{
int memory;
String manufacturer;
RAM(int memory, String manufacturer)
{
this.memory = memory;
this.manufacturer = manufacturer;
}
void displayRAMInfo()
{
System.out.println("RAM Info:");
System.out.println("Memory: " + memory + " GB");
System.out.println("Manufacturer: " + manufacturer);
}
}
}

class CpuDetails
{
public static void main(String[] args)
{
Scanner sc=new Scanner(System.in);
System.out.print("Enter Processor Price");
int price=sc.nextInt();
CPU c1=new CPU(price);

System.out.print("Enter Number of Cores");
int cor=sc.nextInt();
sc.nextLine();
System.out.print("Enter Processor Manufacturer");
String manf=sc.nextLine();
CPU.Processor p1 = c1.new Processor(cor, manf);

System.out.print("Enter Memory");
int mem = sc.nextInt();
sc.nextLine();
System.out.print("Enter RAM Manufacturer");
String manf1 = sc.nextLine();
CPU.RAM r1 = new CPU.RAM(mem, manf1);
```

```
c1.display();
p1.displayProcessorInfo();
r1.displayRAMInfo();
}
}
```

Output

```
its@mits-Veriton-M200-H510:~/gokul java$ java CpuDetails
Enter Processor Price
45000
Enter Number of Cores
8
Enter Processor Manufacturer
Intel
Enter Memory
16
Enter RAM Manufacturer
Kingston
CPU Info:
CPU Price: 45000 RS
Processor Info:
Cores: 8
Manufacturer: Intel
RAM Info:
Memory: 16 GB
Manufacturer: Kingston
```

Experiment 13 Date:

Array of objects

Aim:

Program to create a class for Employee having attributes eNo, eName, eSalary. Read ‘n’ employee information and Search for an employee given eNo, using the concept of array of Objects.

Program

```
import java.util.*;
class Employee
{
int eNo;
String eName;
double eSalary;
Employee(int no, String name,double salary)
{
eNo = no;
eName = name;
eSalary = salary;
}
void display() {
System.out.println("Employee Number: " + eNo);
System.out.println("Employee Name: " + eName);
System.out.println("Employee Salary: " + eSalary);
}
}
class EmployeeSearch
{
public static void main(String[] args)
{
Scanner sc = new Scanner(System.in);
System.out.print("Enter number of employees");
int n = sc.nextInt();
sc.nextLine();
Employee e1[] = new Employee[n];
for (int i = 0; i < n; i++)
{
System.out.print("Enter Employee Number");
```

```
int no=sc.nextInt();
sc.nextLine();
System.out.print("Enter Employee Name");
String name = sc.nextLine();
System.out.print("Enter Employee Salary");
double salary = sc.nextDouble();
e1[i] = new Employee(no, name, salary);
}
System.out.print("Enter Employee Number to Search");
int sNo = sc.nextInt();
int flag=0;

for (int k = 0; k < n; k++)
{
if (e1[k] != null && e1[k].eNo == sNo)
{
flag=1;
System.out.println("Employee Found");
e1[k].display();
break;
}
}
if (flag==0)
{
System.out.println("Employee not found");
}
}
}
```

Output

```
mits@mits-Veriton-M200-H510:~/gokul java$ java EmployeeSearch
Enter number of employees
3
Enter Employee Number
101
Enter Employee Name
gokul
Enter Employee Salary
50000
Enter Employee Number
```

```
102
Enter Employee Name
abhijith
Enter Employee Salary
56000
Enter Employee Number
103
Enter Employee Name
adwaith
Enter Employee Salary
60000
Enter Employee Number to Search
101
Employee Found
Employee Number: 101
Employee Name: gokul
Employee Salary: 50000
```